A Python Wrapper for Coupling Hydrodynamic and Oil Spill Models

Itay Rosenzweig, M.S
Ben R. Hodges, Ph.D.

Center for Research in Water Resources
The University of Texas at Austin

CRWR Online Report 11-09
Submitted to Texas General Land Office Oil Spill Prevention & Response
FY 2011 Report under Contract No. 10-097-000-3928
October 31, 2011

Acknowledgments

The work herein was supported by the Texas General Land Office, Oil Spill Prevention & Response Program, under contract 10-097-000-3928 supervised by Dr. Robert “Buzz” Martin. This contract provided partial support to the National Oceanographic and Atmospheric Administration (NOAA) Office of Response and Restoration, which provided additional support through contributions of the time and knowledge of Dr. Chris Barker. Extensive portions of this technical report are identical to portions of the M.S. thesis of Rosenzweig (2011) that was supported under this contract. For clarity, the prior work is duplicated herein without further citation.

Executive Summary

This report presents a new method for coupling different hydrodynamic models with an oil spill transport model. The hydrodynamic models used are the two-dimensional (2D) TxBLEND model and the three-dimensional (3D) SELFE model. The oil spill transport model is the General NOAA Operational Modeling Environment (GNOME) developed and maintained by the National Oceanic and Atmospheric Administration (NOAA). The new coupling method is an automated approach using Python scripting to eliminate the labor-intensive manual reformatting of hydrodynamic outputs to GNOME inputs. The integrated model system is demonstrated by independently coupling GNOME to TxBLEND and SELFE and modeling a hypothetical spill in Galveston Bay.

Table of Contents

1 Introduction **1**
1.1 Overview *1*
1.2 Oil Spill Modeling *1*
1.3 Models *3*
1.4 Study site – Galveston Bay and the Houston Ship Channel *4*
1.5 Objectives *4*

2 Methodology **6**
2.1 Overview *6*
2.2 Python and Object Oriented Programming *7*
2.3 Approaches to Model Coupling *7*
2.4 Python Wrapper *8*
2.5 Model setup *9*

3 Demonstration simulations **10**
3.1 Overview *10*
3.2 TxBLEND and SELFE 2D hydrodynamics *11*
3.3 TxBLEND and SELFE 2D wind effects *11*
3.4 Importance of GNOME wind at low wind speeds *12*
3.5 SELFE 3D at higher speeds *13*

4 Conclusion **13**

A Appendix: Python Wrapper **14**

B Python Wrapper User Manual **18**
B.1 Introduction *18*
B.2 Operating system *18*
B.3 Python interpreter *18*
B.4 TxBLEND, SELFE and GNOME *18*
B.5 Making the module scripts executable *19*
B.6 File Directory Structure *20*
B.7 Example SELFE simulation *21*

C References **39**

D List of publications developed from this research **42**

This page intentionally left blank for two-sided printing.

1 Introduction

1.1 Overview

This report presents a new approach to automated operation of oil spill and hydrodynamic models for Texas bays as part of an ongoing effort funded by the Texas General Land Office (GLO) in collaboration with the Texas Water Development Board (TWDB) to improve Texas' oil spill response. These agencies conduct operational modeling of the entire Texas coastline, working collaboratively with the National Oceanographic and Atmospheric Administration (NOAA) during oil spill emergencies.

The existing operational model for Texas embayments uses the two-dimensional (2D) TxBLEND hydrodynamic model to simulate water velocities, followed by manual data transfer to the General NOAA Operational Modeling Environment (GNOME) to forecast the oil spill trajectory. The present study develops an automated coupling system based on Python scripting to directly link the hydrodynamic and oil spill models. This coupled system is tested using demonstration simulations comparing GNOME trajectories driven by TxBLEND with those developed using a three-dimensional (3D) hydrodynamic model known as SELFE. The test cases use the entrance to the Port of Houston through Galveston Bay, which is the second busiest port in the United States (Port of Houston Authority, 2011).

1.2 Oil Spill Modeling

Oil spill modeling is a three-stage process. The first stage is input data preparation, the second stage is hydrodynamic modeling, and the third stage is spill trajectory modeling (Wang et. al., 2005). Input data preparation requires identifying all the operational parameters of the two models, including the model domains and grids, tidal boundary conditions, inflows, meteorological data, and initial hydrodynamic conditions. The hydrodynamic model predicts water velocities based on input data; the spill trajectory model applies the predicted velocities and other environmental forces to simulate the oil particles' fate and transport. In practice, the environmental forces (i.e. other than water velocities) applied by the spill trajectory model are wind, diffusion, and weathering/decay (Beegle-Krause, 2001).

The general workflow for oil spill modeling is illustrated in Figure 1. Some authors group the steps differently (e.g. Kerbaol & Collard, 2005) and some models (have built-in visualization software (e.g. GNOME, see Beegle-Krause, 2001).

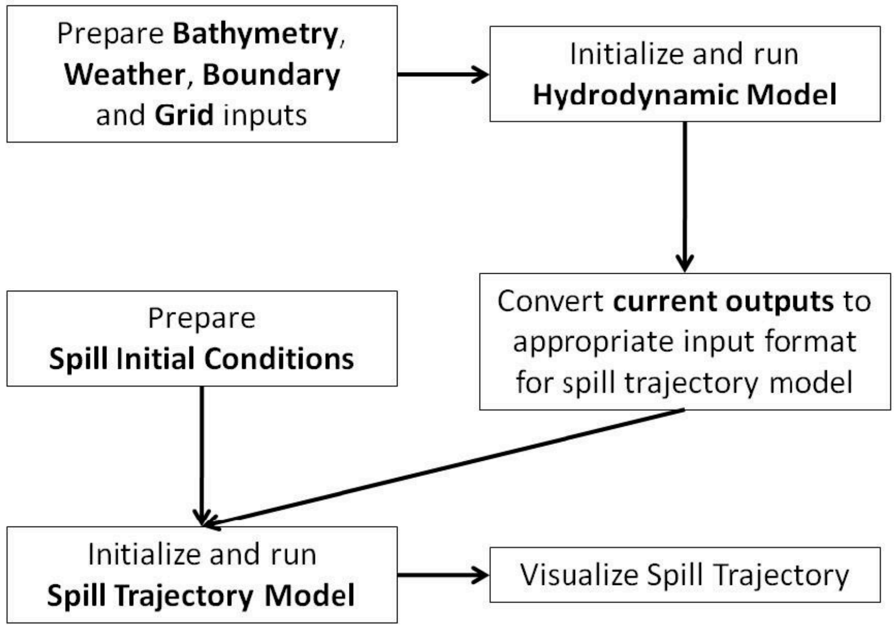


Figure 1. Workflow for oil spill modeling.

The workflow in Figure 1 presumes uni-directional coupling; i.e. a spill is affected by the hydrodynamics, but the hydrodynamics are unaffected by the spill. However, unlike many transport problems, oil on the water surface is not a simple passive tracer moving at the water velocity. Instead, it forms a surface layer over the water that interacts with both water and atmosphere, which may impede the development of wind-driven currents (ASCE Task Committee, 1996). Such feedback effects would require either close coupling of the hydrodynamics and oil spill model, or an iterative looping procedure. Unfortunately, with the present state-of-the-art, the feedback effects from the oil spill to the hydrodynamics cannot be modeled. Fortunately, for the smaller spatial scales and short durations associated with oil spills in a confined bay, this issue is of limited concern. Thus, the modeling in this project retains the standard uni-directional coupling paradigm.

Although surface oil spill trajectory models (e.g. GNOME) use only the 2D surface water velocity field, accurate representation of the surface currents arguably requires accurate representation of the near-subsurface velocities, which may be different than the depth-averaged velocities (Wang & Shen, 2010). Studies of hypoxia in Corpus Christi Bay indicate that TxBLEND does not accurately predict behavior in the subsurface (Furnans, 2004). A 3D model allows subsurface layers to interact with, but behave separately from, the surface, which may more accurately represent near surface water velocities.

Presently, the operational hydrodynamic model for Texas oil spill response is TxBLEND as discussed in Crocket (2010). Wind data driving the hydrodynamics are obtained from the Eta Model from the National Centers for Environmental Prediction (NCEP). The Eta Model is a gridded reanalysis providing wind forecasts and hindcasts, the latter using interpolated field observations (Black, 1994). Tide forecasts for TxBLEND use tidal harmonic constituents. Tide hindcasts use field observations from the Texas Coastal Ocean Observation Network (TCOON)

1.3 Models

1.3.1 GNOME

GNOME is NOAA's in-house oil spill trajectory model. NOAA is responsible for providing information to promote safe and effective response to all oil spills in US surface waters (NOAA OR&R, 2011). Along with state and local agencies, NOAA has applied GNOME over a range of regimes, from inland lakes to the open ocean. The model is used internationally (Basar et. al., 2006) and can be considered a *de facto* standard for operational oil spill modeling.

GNOME is a Lagrangian particle-tracking model that transports simulated oil particles called Lagrangian Elements (LE). The LE are driven by water currents, wind, diffusion and decay (Beegle- Krause, 2001). The standard GNOME application uses a built-in graphical user interface (GUI), which cannot be easily automated in a coupled workflow. However, the GUI can be bypassed and the GNOME model configured through using a standard command-line interface or a command file of ASCII text, which can be automated.

1.3.2 TXBLEND

TXBLEND is a 2D depth-averaged, unstructured grid model solving the volume conservation, the hydrostatic momentum equations, and the advection-diffusion equations for salinity transport. The model has been used extensively for modeling Texas bays, e.g. Powell et al. (1997).

1.3.3 SELFE

SELFE is a Semi-Implicit, Eulerian-Lagrangian, Finite Element model, which uses a 3D horizontally-unstructured grid with hybrid sigma-Z vertical coordinates for vertical discretization (Zhang & Baptista, 2008). Although developed for the Columbia River estuary, the model has since been successfully applied to the Chesapeake Bay (Gong et. al., 2009), the Ria de Aveiro in Portugal (Rodrigues et. al., 2009) and the Guadiana Estuary (Oliveira et. al., 2006). SELFE uses the Generic Length Scale turbulence closure of Umlauf and Burchard (2003), which encompasses the standard κ-ε, κ-ω, and Mellor-Yamada 2.5 closure.

1.4 Study site – Galveston Bay and the Houston Ship Channel

Texas bays are a challenge for hydrodynamic modeling, which affects forecast reliability for oil spills. In this project, we will test the coupled modeling system on Galveston Bay, which like many Texas bays, is relatively flat and shallow (3-4 m depth). As illustrated in Figure 2, narrow ship channels serving the commercial harbors are dredged to approximately 15 m deep and 120 m wide (Department of Commerce, 1998). The path of the Houston Ship Channel through Galveston Bay is shown in Figure 3.

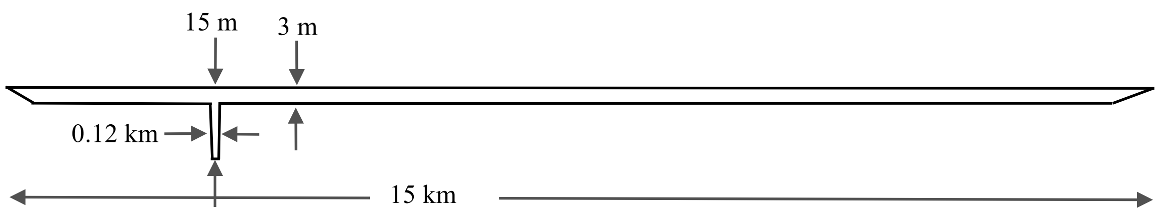


Figure 2. Schematic representation of typical cross-section of a Texas bay with a 60:1 ratio of horizontal:vertical scales.

Furnans (2004), Kulis and Hodges (2006) and Pothina (2009) all showed that 2D models have difficulty representing hydrodynamics of Texas bays because of the narrow, deep ship channels. Furnans (2004) and Pothina (2009) both showed how the 2D depth-averaged velocity in the channel limits cross-channel fluxes. Furnans (2004) studied circulation through known hypoxic regions in Corpus Christi Bay, illustrating the isolation effects of the ship channel as shown in Figure 4. Pothina (2009) used vertical grid resolution studies with SELFE for a geometrically idealized ship channel to examine cross channel currents on the surface above the channel, as shown in Figure 5. Complementing these model studies, a recent field study at the Aransas Pass indicates that currents above a ship channel can have a significant cross channel component (Min, 2010).

1.5 Objectives

The primary objective of this study is to develop a Python "Wrapper" coupling the hydrodynamic models TxBLEND and SELFE with GNOME. The Python Wrapper minimizes the manual steps required to run and couple the models by ensuring any necessary manual steps can be completed prior to starting a model run. The system is demonstrated using a series of simulations of TxBLEND and SELFE coupled with GNOME for a hypothetical spill in Galveston Bay. These simulations are used as preliminary investigations into modeling requirements for nowcast/forecast system for Texas bays to evaluate the sensitivity of hydrodynamic modeling results to the choice of model, dimensionality and wind effects.

Figure 3. Satellite image of Houston Ship Channel (Google Earth, 2011)

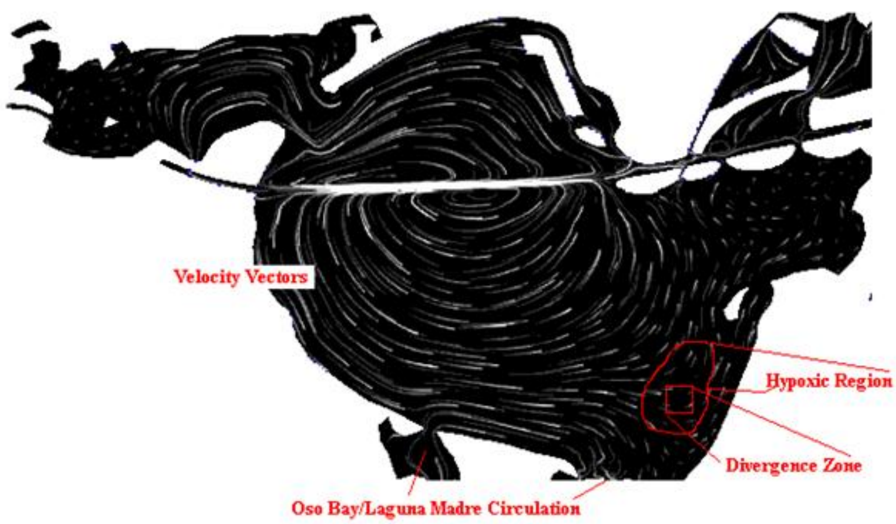
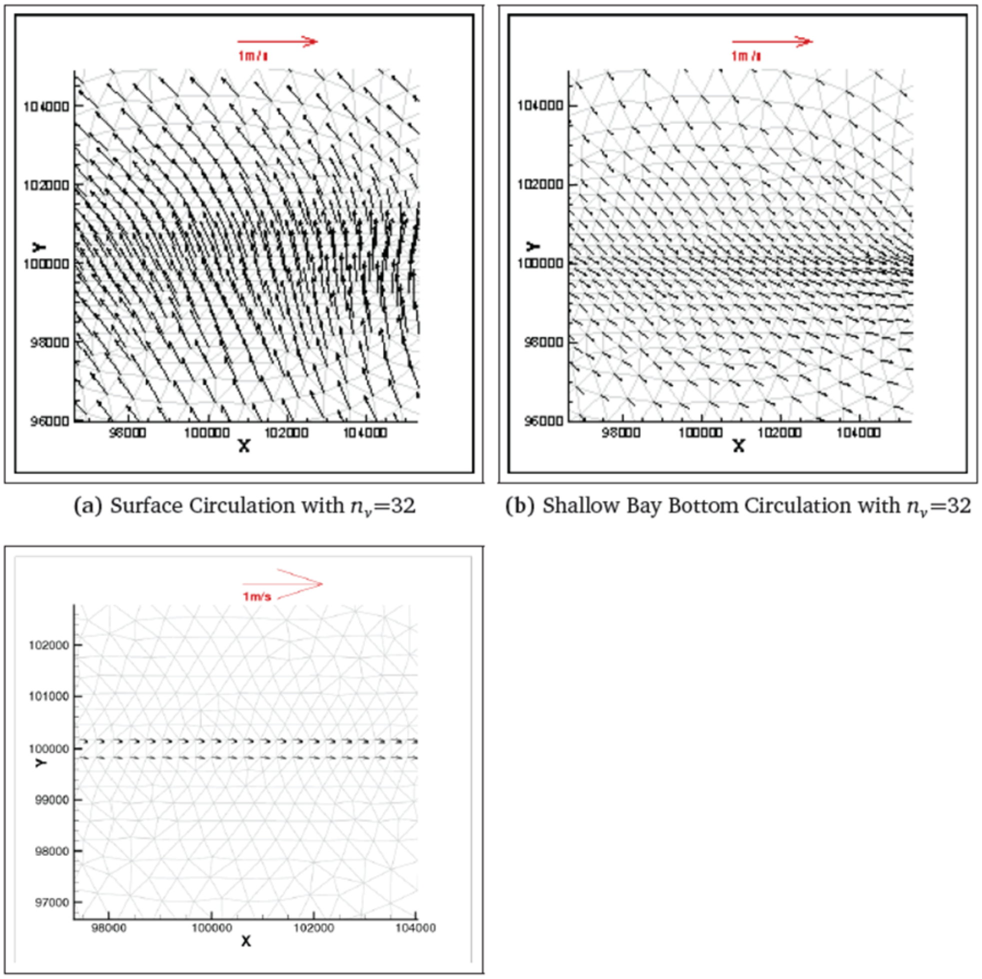


Figure 4. Depth-averaged (2D) model of circulation in Corpus Christi Bay, from Furnans (2004).



(a) Surface Circulation with n_v=32 (b) Shallow Bay Bottom Circulation with n_v=32

(c) Ship Channel Bottom Circulation with n_v=32

Figure 5. Modeled 3D flow patterns in the vicinity of a ship channel, from Pothina (2009); a) circulation at the water surface, b) circulation at the bottom of the shallow bay, and c) circulation in the deep ship channel.

2 Methodology

2.1 Overview

The GNOME graphical user interface was designed for stand-alone application, so that users familiar with GNOME could take velocity fields from any hydrodynamic model and develop an oil spill simulation. Although this approach is ideal when hydrodynamic and oil spill modeling are handled by separate organizations, the GUI is less desirable when a single agency is conducting the modeling. A seamless coupling between hydrodynamic and oil spill models without intermediate user intervention simplifies nowcast/forecast operational use. In this project, a scripting language (Python) is used to build an automation wrapper that uses the GNOME command file rather than the GUI to configure and execute GNOME. The Python Wrapper allows outputs of the hydrodynamic model to be processed and input directly into GNOME without user intervention.

2.2 Python and Object Oriented Programming

Python is an open-source programming language that emphasizes modularity, readability, and execution speed. It is a "just-in-time" compiled language, which means code is not separately compiled by the user before running. The translation between the Python syntax and machine language is part of the run-time process. Python and similar languages are sometimes called "uncompiled" languages (Lutz, 2011). Python has two features that are invaluable for the present work: 1) Python is a scripting language, which means it can open files, create and write files, save files, and run other applications through a computer's operating system; and 2) Python uses modularity, which simplifies wrapping the workflow for multiple models.

In the present work, a "module" is an object that performs one set of functions. Objects are allowed to interact with each other through a set of rules defined by the master wrapper, which is a straight-forward application of object oriented programming (Mitchell, 2003). An advantage of this approach is that each code section is an independent module that can be switched out with ease, usually just by changing a single line in the master wrapper. Thus, making the TxBLEND wrapper work with the SELFE hydrodynamic model does not require an entire new program, just swapping a few modules. Future implementation of other hydrodynamic models is thereby simplified. More speculatively, the wrapper approach might be used for iterative model coupling to represent wind forcing attenuation in the hydrodynamic model due to presence of the oil spill (discussed in §1.2). A further advantage is the rules that define how the objects interact with each other prevent bugs in one section of code from "leaking" into other sections, reducing error checking and debugging times over traditional programming techniques (Lutz, 2011).

2.3 Approaches to Model Coupling

Model systems integrating different component models can be tightly coupled, coupled through an interface, or loosely coupled. In tightly coupled systems, the model integration requires rewriting one or more models to directly incorporate others as submodels within the same code and data structure. Such tight model coupling customizes integration of individual models, which usually translates to efficient computational implementation. The submodels exchange information without the inefficiencies of writing to a file, as is typically used in looser forms of model coupling. A disadvantage is that tight coupling increases model development time and complicates debugging. Examples of tightly coupled systems are the Mueller et al. (2011) ocean circulation model coupled with sea ice, and the Fach & Klinck (2006) coastal ocean and particle tracking system.

In contrast to direct integration through tight coupling, to use interface coupling models need only conform to a standard input/output format such as OpenMI (OpenMI Association, 2011). This approach provides an environment where different models may exchange data through a single standard interface. However, unless models are initially developed in compliance with the interface standard, significant code rewriting is generally required. As examples, OpenMI was used by Bulatewicz et. al. (2010) to integrated agricultural, groundwater, and economic models and by Christensen (2004) for an integrated river basin management and hydrological model.

Loosely coupled models exchange data through their originally-designed input and output formats, requiring a conversion step to move data from one model to the next. This approach is generally less efficient than tight or interface coupling, but is easier to develop and debug. Loose coupling is desirable when the individual model codes are developed and maintained by different organizations that use different interface standards. Loose coupling can also be an efficient way of testing and developing model coupling ideas prior to undertaking rewriting the underlying model codes for either interface or tight coupling.

Because GNOME, SELFE and TxBLEND have been developed by three different organizations, the present work uses a loose coupling approach through a Python Wrapper.

2.4 Python Wrapper

The Python Wrapper automates the workflow (q.v. Figure 1) with the exception of input data preparation. At this time, all input data files (including the GNOME command file), must be prepared manually in advance of a model run. The modules are operated by a master wrapper that call the operational modules in the correct order. The structure of the Python Wrapper is shown in Figure 6.

The first wrapper module checks that necessary input data are available. Missing files result in an error condition, ensuring that time is not wasted by running models that will simply crash due to lack of data. The second module calls the hydrodynamic model executable (either SELFE or TxBLEND). Output files are automatically saved to the same directory that contains the model executable. The third module converts the outputs from the hydrodynamic model to GNOME input and saves the new GNOME input file with the name expected by the GNOME command file. Because each hydrodynamic model has different input, executable, and formatting, separate module versions are required for each hydrodynamic model. The fourth module calls GNOME. This module that is identical for all hydrodynamic models.

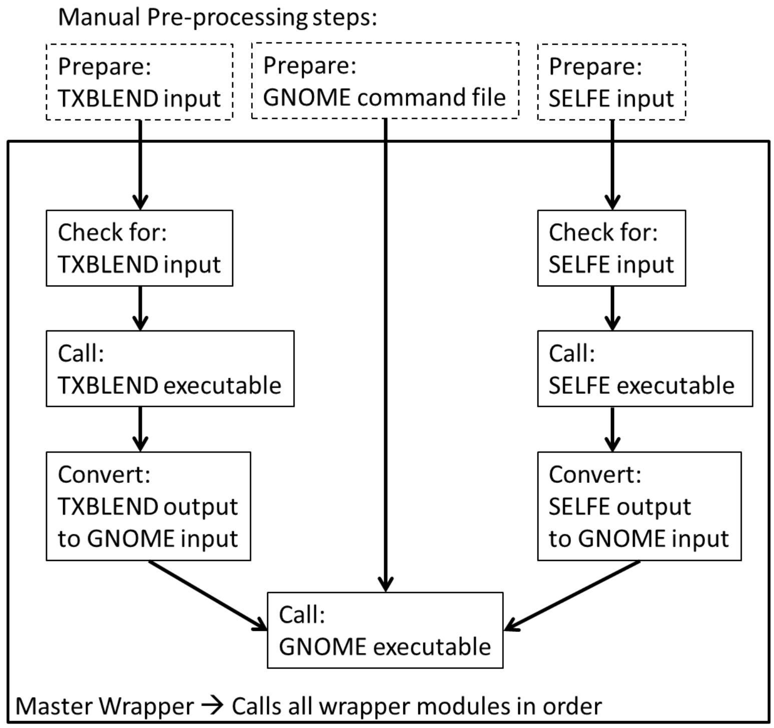


Figure 6. Python Wrapper modules and structure

These modules automate the entire modeling process from the hydrodynamic modeling to the trajectory simulation. Spill visualization is accomplished using GNOME's visualization capability. The only manual steps are preparation of the input files for the hydrodynamic model and the preparation of the GNOME command file.

The Python Wrapper scripting approach is provided in Appendix A. A user guide to the Python Wrapper is provided in Appendix B.

2.5 Model setup

The Python Wrapper has been tested by applying the coupled SELFE/GNOME and TxBLEND/GNOME model combinations in several scenarios. This effort is used as a preliminary investigation into the similarities and differences in oil spill predictions using different modeling approaches. The models are compared in dimensionality (2D vs. 3D) and response for three wind conditions (none/low/high) for the six simulations listed in Table 1.

TxBLEND and 2D SELFE were run for 48 hour simulations, coupled to GNOME for an imaginary100 barrel point-source spill in the middle of the Galveston Bay ship channel. The model grids were identical, as provided by the TWDB (D. Pothina, pers. comm.). Tidal forcing for the offshore boundary is simple diurnal sinusoid using an amplitude of 0.3 m. The low and high wind conditions are illustrated in Figure 7, based on the NCEP Eta Model hindcast wind conditions (TAMU, 2010) for the first week in April (low wind) and the first week in May (high wind). The average wind speed during the low wind conditions is 3.8 kts and during the high wind condition is 10.2 kts.

Table 1. Simulations

ID	Hydrodynamic Model Wind	GNOME Wind	Model
A	none	none	2D TxBLEND
B	none	none	2D SELFE
C	low	low	2D TxBLEND
D	low	low	3D SELFE
E	none	low	3D SELFE
F	high	high	3D SELFE

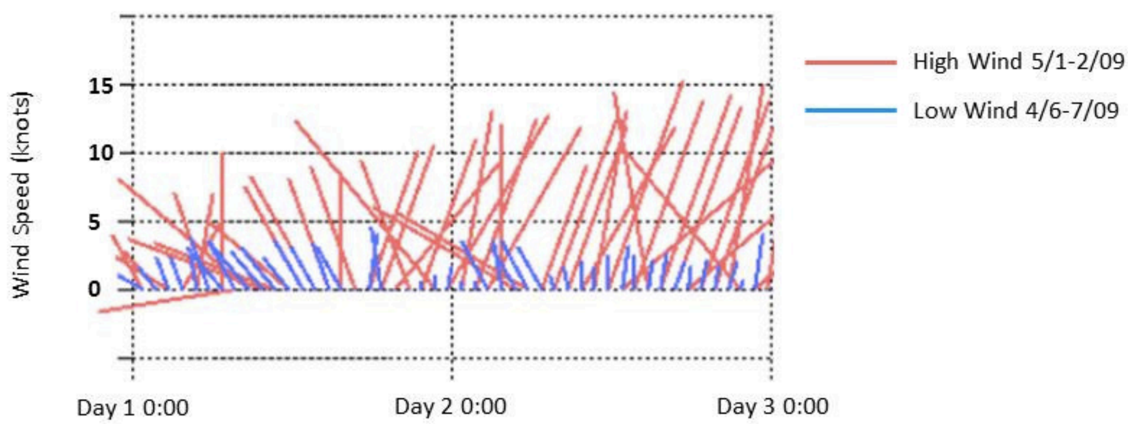


Figure 7. Wind conditions

Although SELFE is a 3D model, it can be run in a 2D mode using only a single vertical grid layer, thus providing a direct comparison of the wind-forcing and tidal algorithms in the two models. For the 3D SELFE simulations, 10 sigma-layers were used.

3 Demonstration simulations

3.1 Overview

The Python Wrapper was applied to run SELFE/GNOME and TxBLEND/GNOME for the six simulations outlined in §2.5. In the following sections, we compare the final spill location predicted by GNOME for the different model forcing conditions.

3.2 TxBLEND and SELFE 2D hydrodynamics

Comparing results Simulations A and B provides insight into the underlying tidal hydrodynamic algorithms of the two models. The wind is zero both for hydrodynamic forcing and the oil transport in GNOME. As shown in Figure 8, using the same 2D grid with identical tidal forcing provides very similar results. Because TxBLEND and SELFE use different governing equations and different numerical algorithms, these result sprovide confidence that the basic depth-averaged hydrodynamics are being correctly computed.

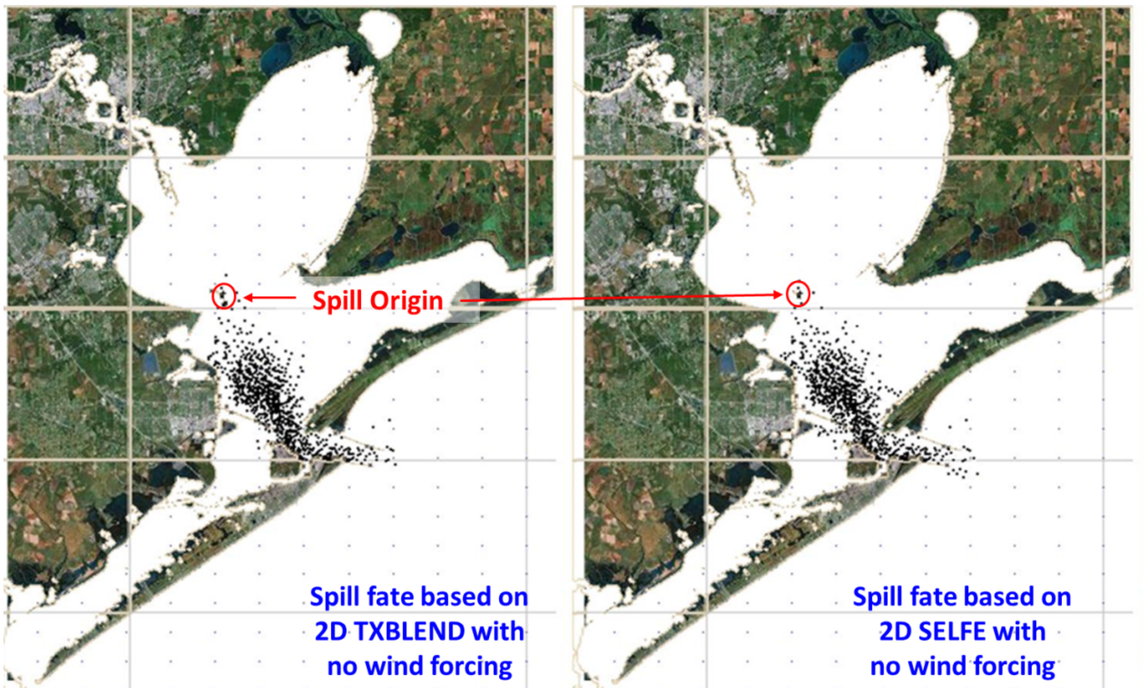


Figure 8. Simulations A (left) and B (right) for 2D models without wind.

3.3 TxBLEND and SELFE 2D wind effects

TxBLEND and SELFE use different approaches to including wind driven currents. The TxBLEND approach is an empirical model (unpublished), whereas SELFE used the Generic Length Scale turbulence closure (Umlauf and Buchard, 2003) to provide the wind effect on momentum in the water column. Figure 9 provides a comparison of the 2D models using low wind conditions. Because GNOME for both models also uses direct wind forcing on the oil spill, it is not surprising that the models show qualitatively similar effects. However, there are quantitative differences are clearly important. The SELFE model predicts landfall of the spill along the northern edge of the bay, whereas the TxBLEND model shows the spill clustering towards the eastern shore. Thus, differences between the wind momentum approach in the two models cannot be neglected.

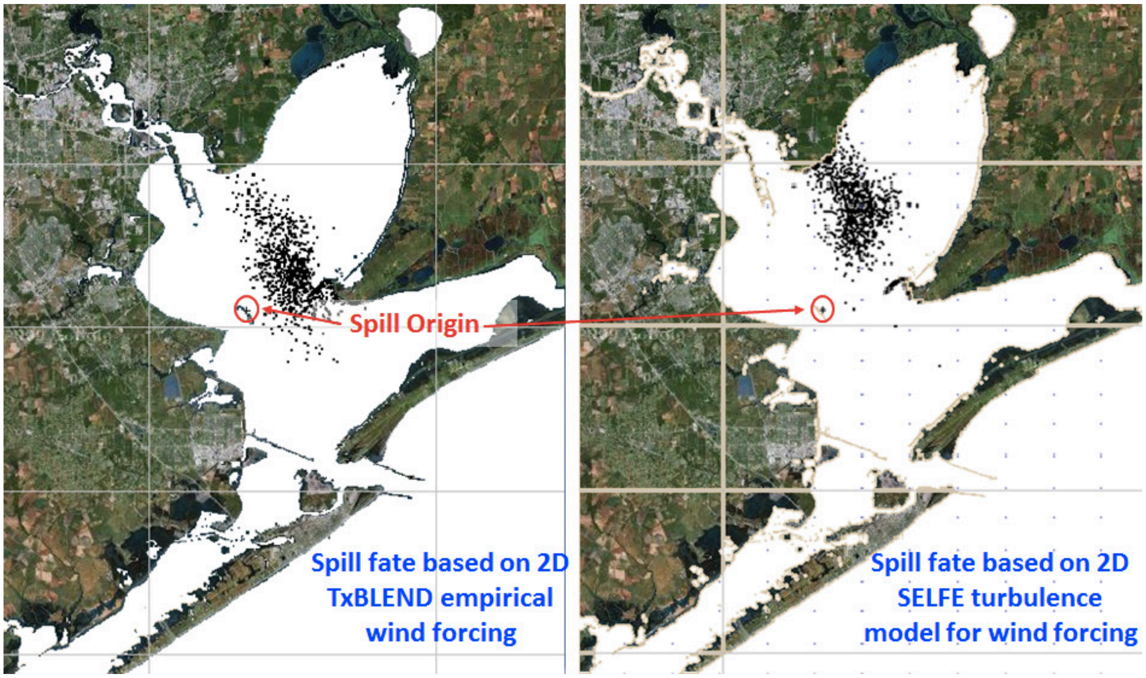


Figure 9. Simulations D (left) and E (right) for low wind conditions in 2D TxBLEND and 2D SELFE.

3.4 Importance of GNOME wind at low wind speeds

In Figure 10, results are shown for SELFE in 2D and 3D with the low wind condition, and in 3D when the wind is turned off in hydrodynamics but maintained at the low wind condition in GNOME. For these results (as well as in Figure 9), the GNOME wind model clearly has a dominant effect.

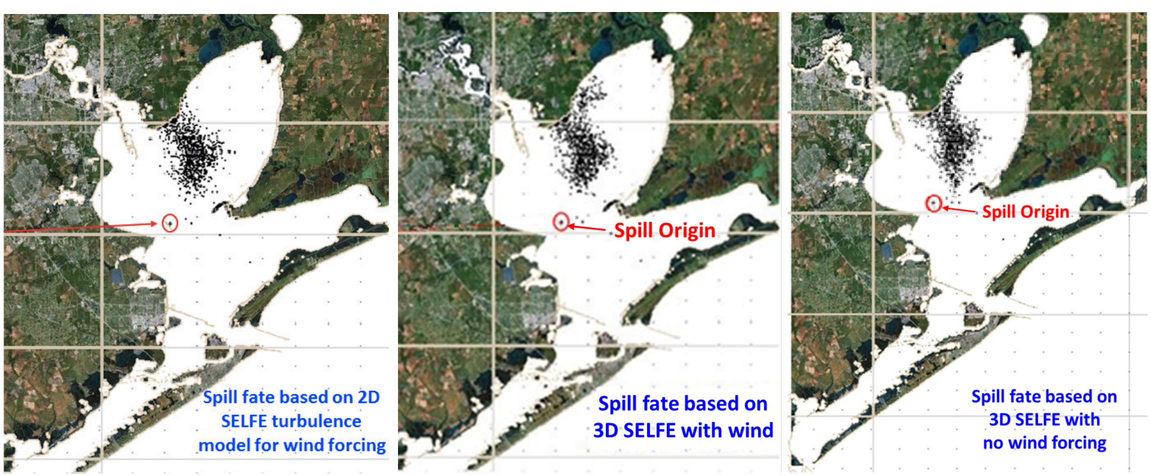


Figure 10. SELFE results for simulation B (left), D (center) and E (right) showing qualitatively similar results.

3.5 SELFE 3D at higher speeds

Comparing Simulations D and F for low and high wind conditions using SELFE 3D, as shown in Figure 12, provides confidence that wind in the 3D SELFE/GNOME combination is affecting the water circulation and therefore the oil spill fate.

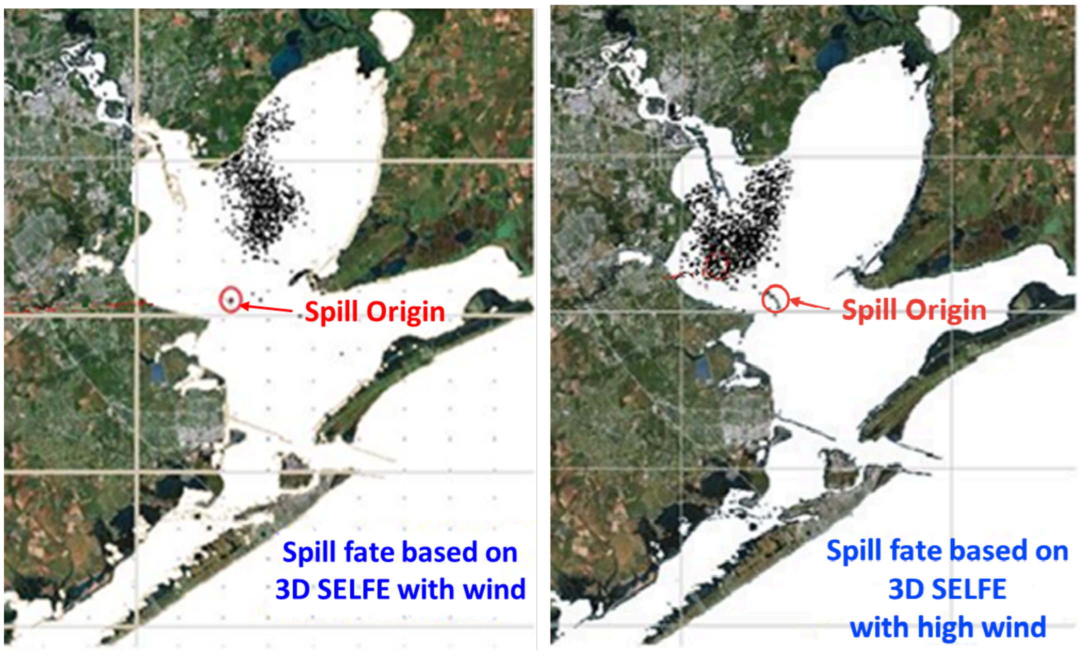


Figure 12. Simulations D and F for SELFE 3D with low wind (left) and high wind (right).

4 Conclusion

The Python Wrapper successfully automates the link between TxBLEND and SELFE hydrodynamic models and the GNOME oil spill transport model. In the present system, the hydrodynamic model input files and a GNOME command file must be manually prepared before the coupled models can be run.

The demonstration simulations in §3 indicate that the difference between 2D and 3D simulations for shallow Texas bays may be significant for predicting oil spill fate. Although TxBLEND and SELFE 2D models are essentially indistinguishable without wind forcing, the addition of wind and using 3D makes substantial changes in the results.

To continue the development of a fully automated system for nowcasts/forecasts the Python Wrapper should be linked to automatic data collection. Presently, TWDB is running TxBLEND nowcasts with automated tidal and inflow data (D. Pothina, pers. comm.). This system could be extended to include wind forcing and use Python scripting to prepare input files for either TxBLEND or SELFE building on the PythonWrapper..

A Appendix: Python Wrapper

This appendix provides the code and structure of the Python Wrapper. S ections where the code is simple are reproduced in their entirety and annotated. More complex sections where the code is very lengthy or the functionality of the code is not immediately obvious are presented as psuedocode with explanation. T he code shown is for a TXBLEND run. Note that exactly corresponding modules exist for SELFE runs. Places were the differences are important are explicitly noted.

MODULE 0: Master Wrapper

```
# master_wrapper.py
# This is the master wrapper which calls all the other
# modules.

# This first section imports all of the functional
# modules. It allows the master wrapper to call all
# of the functions within the modules

     import txblend_in          # or selfe_in
     import run_txblend         # or run_selfe
     import txblend_to_gnome    # or selfe_to gnome
     import run_gnome

# This second section runs the functions defined in the
# imported modules.

     txblend_in.check()
     run_txblend.run()
     txblend_to_gnome.main()
     run_gnome.run()
```

MODULE 1: Checking Input Files

```
# txblend_in.py
# This module checks for all TXBLEND input files.

     def check()  # Creates a function that the master
               # wrapper can call on

     open(__list of TXBLEND input files__, ‘r’)

               # Opens all the TXBLEND input files
               # in ‘read only’ mode

     filename.close()
```

```
                    # List all the files with the
                    # '.close()' argument.

# The logic behind this segment is that the function
# cannot open a non-existent file in 'read only' mode.
# If the necessary file does not exist and the wrapper
# fails to open it, the entire program will stop and
# display an error.
```

MODULE 2: Running Hydrodynamic Model

```
# run_txblend.py
# This module runs the TXBLEND executable.

# The Python library module 'subprocess' is used to
# write commands that mimic the terminal interface.

     def run()

          import subprocess
          subprocess.Popen('txblend.exe')

               # This format will only work if the
               # model executable is located in the
               # user's $PATH environment variable.
               # Else, include the entire path to
               # the executable.
```

MODULE 3: Converting Hydrodynamic Model Output to GNOME Input

```
# txblend_to_gnome.py
# This module converts TXBLEND current outputs to GNOME
# inputs.

     def main()

     import netcdf4 as nc

               # Library of tools which allows
               # Python to write in netCDF format

     file = open(__TXBLEND output file__, 'r')

     file2 = open(__GNOME current file__, 'w')

               # This opens GNOME's current input
               # file in 'read/write mode'. If the
```

```
          # file does not exist, it will be
          # created.

=========//===========

file2.write(...)

     # This section writes in the header for
     # GNOME current input file.
     # See [Beegle-Krause, 2001] for details.

=========//===========
```

```
# Instead of requiring index arguments for an iterative
# argument, Python can iterate over any object. In this
# case, the wrapper iterates the 'for' loop over every
# 'line' in the object 'file'.

     for line in file:

               thisline = str(line)

                    # Converts the contents of the
               # line into a string object

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
       IMPORTANT DIFFERENCE BETWEEN TXBLEND AND SELFE
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

FOR TXBLEND:   if (TXBLEND output format)in thisline:

               # This statement searches for the
               # string with TXBLEND current output
               # in the output file. This is unique
               # to TXBLEND where every line in the
               # output file which has current data
               # has the string 'u ='.

FOR SELFE:     if (SELFE output format) in thisline:

               # In SELFE, one of the velocity
               # variables is called UVEL, so
               # 'UVEL =' in a line means that
                    # line contains current data.
                    # However, multiple vertical layers
                    # have current data stored, and only
                    # the surface layer is required.
```

```
                    # This is identified by the
                    # ‘SLAYER = 0’ string.

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
                  END IMPORTANT DIFFERENCE
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

               ============//============

               # This segment translates ‘thisline’
               # with the current data into a line
               # in the proper format for GNOME.
               # Using the “copy”, “paste”, “find”
               # and “replace” Python functions,
               # the variables  on the line can be
                    # renamed and reordered in GNOME’s
                    # expected format.

               ============//============

     file2 = __GNOME current file__.nc

               # This step saves the text file
                    # Python has been writing to as a
                    # netCDF file that GNOME can read.

     file.close()
     file2.close()
```

MODULE 4: Running GNOME

```
# run_gnome.py
# This module runs the GNOME executable
# and saves an screen capture of the final spill map

   def run()

     import subprocess

     subprocess.Popen(‘./gnome.exe’)

# The './' command ensures that the copy of the GNOME
# executable in the working directory is called.
# This is necessary to ensure that GNOME identifies the
# prepared command file.
```

B Python Wrapper User Manual

B.1 Introduction

This appendix is the User's Manual for the Python Wrapper developed to automate the linkage between the National Oceanographic and Atmospheric Administration's (NOAA) General NOAA Operational Modeling Environment (GNOME) and the TXBLEND and SELFE hydrodynamic models. The wrapper was developed as part of GLO Contract 10-097-000-3928. This User's Manual is current as of September 2011.

B.2 Operating system

The Python Wrapper was developed on an Intel Mac running Mac OS X 10.6.6. Mac OS X is the only platform on which TXBLEND, SELFE, and GNOME were all able to be successfully built. However, the Python code itself is platform independent, so if successful Windows builds of SELFE, or Linux builds of GNOME become available in the future, the Python Wrapper can be implemented on those systems without alterations.

B.3 Python interpreter

The Python Wrapper is written in Python 2.6.1. Future Pythong development is moving to the newer Python 3.X interpreter, which is not fully backwards compatible with Python 2.X code. However, there are a number of important existing standard Python libraries which were not available in Python 3.X. These include the python-netCDF library, as well as the Numpy/Scipy libraries. The Wrapper currently requires the netCDF tools, and it is anticipated that further development and extension of the Wrapper in the near future will require other libraries that are not yet ported to Python 3.X.

B.4 TxBLEND, SELFE and GNOME

Fortran source code for the TXBLEND model implemented by the wrapper is available at: https://webspace.utexas.edu/ir926/TXBLEND/

The Wrapper implements SELFE Serial Version 1.5k7. Although a parallel version of SELFE is available, and is recommended by the developers, the serial version is used in this work because the project largely builds on results and insights from the dissertation work of Dharhas Pothina, PhD. His work was done using serial SELFE, so a decision was made early in the development to continue using the serial code. Future development may want to strongly consider moving to the parallel code. Source for the SELFE model is available at: http://www.stccmop.org/CORIE/software/selfe/recent.html

Successful implementation of SELFE requires the netCDF4 and HDF5 C libraries available respectively at: ftp://ftp.unidata.ucar.edu/pub/netcdf and ftp://ftp.hdfgroup.org/HDF5/current/src

To allow the Python Wrapper to interact with SELFE output, the python-netCDF library must be available. In addition to the netCDF4 and HDF5 C libraries already required for SELFE, the Python library depends on the standard Numpy array module available at: http://numpy.scipy.org

The latest Mac build of GNOME is available at: http://response.restoration.noaa.gov/type_topic_entry.php?RECORD_KEY%28entry_topic_type%29=entry_id,topic_id,type_id&entry_id(entry_topic_type)=290&topic_id(entry_topic_type)=1&type_id(entry_topic_type)=3

Note that successful running of the Python Wrapper depends on the individual system it is being run on having all of these dependent models and libraries. Unfortunately, installation of these models and libraries is machine-specific, and at the present stage of development, they all need to be installed individually on each new machine desiring to run the wrapper.

B.5 Making the module scripts executable

The first line in each of the module scripts is " #! /usr/bin/env python ". This line tells the script where in the directory tree the Python interpreter is located. /usr/bin/env is fairly standard for Python installations on UNIX based systems, but if that path is incorrect for the particular machine being used, this line must be manually changed in all the module *.py files. With this header line, each module can be made executable from the command line. At the command line interface (the Terminal.app on Mac), the following command will make the module 'module.py' executable.

```
$ sudo chmod +x /path/to/module.py
```

The 'sudo' command will require a system administrator password to be input before the command is carried out. Executing this command will allow the wrapper modules to be run from the command line (outside of the Python interpreter) by using the command

```
$ /path/to/module.py
```

To simplify this, all the module scripts can be stored in a single directory whose location can be added to the user $PATH environment variable. This can be done on Mac using the Terminal.app command

```
$ PATH=$PATH\:/path/to/where_modules_are_stored
```

Note that the spacing and slash-directionality are important. Executing this command allows the modules to be run from the command line using the command

```
$ module.py
```

The above approach eliminates having to remember and specify the path to where the module is stored.

If the above method is not used, the wrapper can still be run from within the native Python interpreter interface. However, it is more difficult to navigate the directory tree within the interpreter interface. Ease of directory tree navigation – a feature of the command line interface – becomes especially important when scripting an automated run of several instances of the wrapper, each in its own directory for organization.

B.6 File Directory Structure

The Python Wrapper is hardcoded to expect specific file names and locations. The particular working directory where the run's input files are stored must be changed in the module code. Below is shown in list form the directory structure that the wrapper expects.

```
RUN_MASTER_DIRECTORY
    SELFE
      selfe.exe         # a copy of the SELFE executable
      hgrid.gr3         # horizontal grid file
      vgrid.in          # vertical grid file
      param.in          # model run parameter instructions
      wind.th           # wind record
      elev.th           # tidal boundary condition
        outputs         # SELFE expects an outputs directory in the
                        # same location as the inputs
                        # This directory will be populated by SELFE,
                        # not the user
          mirror.out    # runtime output information
          hvel.64       # horizontal velocity output - this is the file
                        # the wrapper needs
                        # any other outputs the user specifies in
                        # param.in
    GNOME
      gnome.exe         # a copy of the GNOME executable
      map.bna           # specifies the outer limits of the modeled
                        # area
      wind.wnd          # wind record, same data but in a different
                        #format than wind.th
      command.txt       # this is the GNOME command file
```

If TXBLEND is used, the same directory structure applies, except instead of a SELFE directory under the master directory, a TXBLEND directory containing a copy of the TXBLEND executable and TXBLEND's lone input file should be used, as shown below:

```
 TXBLEND
txblend.exe
input.txt
output1.txt    # TXBLEND output files will be generated in this
               # directory. The wrapper will only look for this one.
```

This directory structure is designed to organize all the run data in a consistent manner which also satisfies the requirements of the models called by the wrapper. At present, the directory structure is hardcoded into the wrapper. Future development of the wrapper should attempt to eliminate this requirement, as it limits flexibility in the organization of the data. If, instead of being hardcoded into the wrapper, the working directory was a user generated input, the only limits to flexibility of data organization would be the expectations of the models called by the wrapper.

B.7 Example SELFE simulation

To test that the wrapper is working properly with all parts installed correctly, a set of example files is provided for an idealized circular bay with a tidal inlet. The test bay geometry is shown in Figure B.1. This simple bay model runs for 48 hours and is driven by a sinusoidal diurnal tide and a sinusoidal diurnal westerly wind, illustrated in Figure B.2. The tide has an amplitude of 0.3 meters +/- mean sea level, and the wind has an amplitude of 5 m/s. The bay is discretized in a triangular unstructured grid with 43 nodes and 56 elements as shown in Figure B.3. X and Y coordinates are in the standard meters required by SELFE. Input files for this system are provided following the figures.

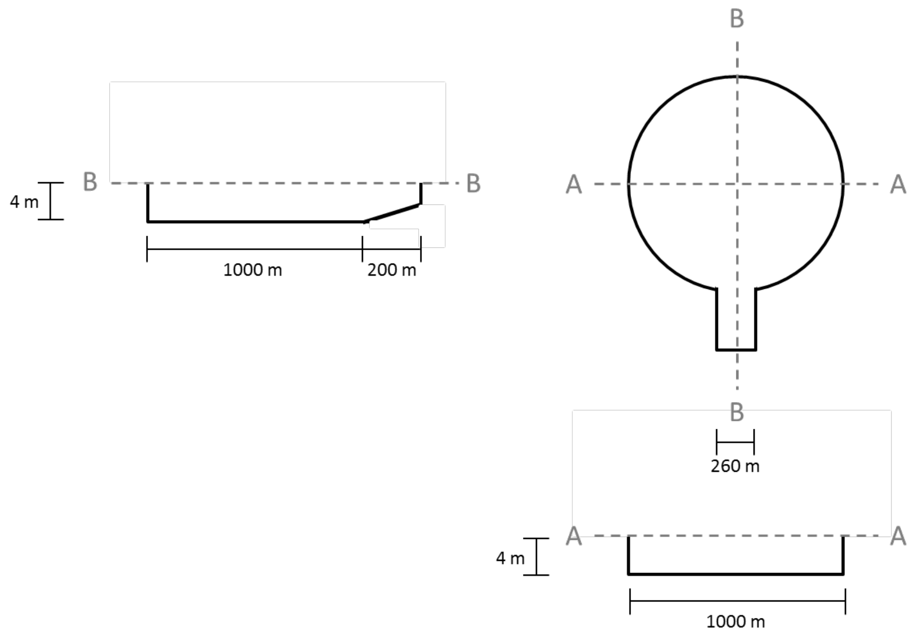


Fig. B.1 Geometry of test bay

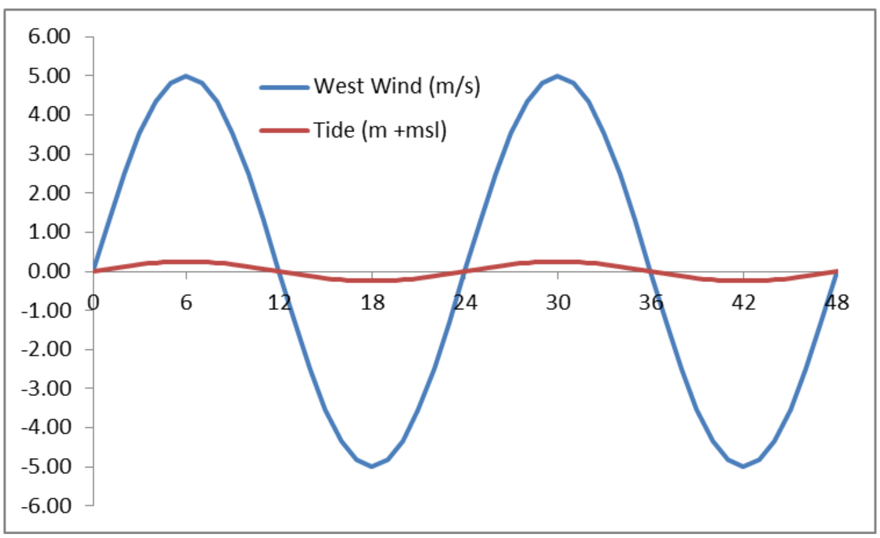


Figure B.2. Wind and tide forcing

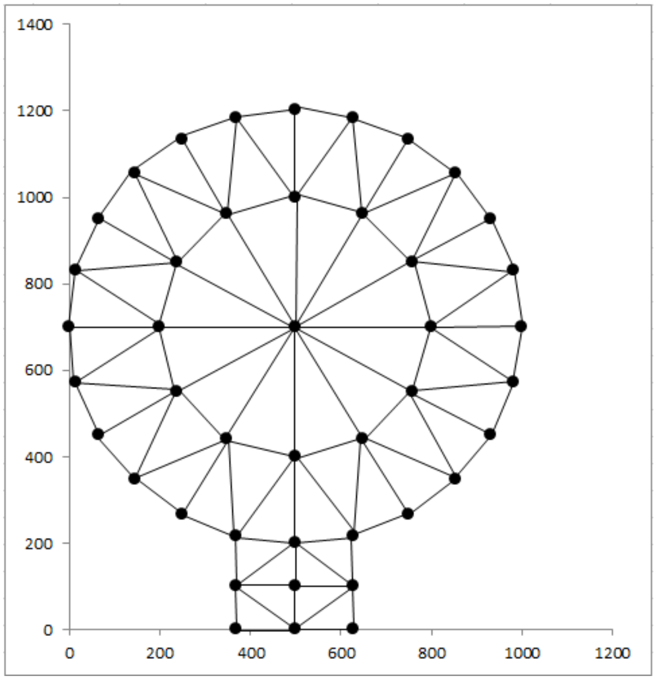


Figure B.3. Grid for test bay

Example input files

The comment lines indicating START OF FILE and END OF FILE should not be included in the files when copying to a text editor. To use these files to test the wrapper, copy the text between the START OF FILE and END OF FILE comments, and save as text files in the directory structure indicated in Fig. 3.2.1.

===START OF HGRID.GR3 FILE===

```
hgrid.gr3
56 43
1   500.00  200.00   4.0
2   370.59  217.04   4.0
3   250.00  266.99   4.0
4   146.45  346.45   4.0
5   66.99   450.00   4.0
6   17.04   570.59   4.0
7   0.00    700.00   4.0
8   17.04   829.41   4.0
9   66.99   950.00   4.0
10  146.45  1053.55  4.0
11  250.00  1133.01  4.0
12  370.59  1182.96  4.0
13  500.00  1200.00  4.0
14  629.41  1182.96  4.0
15  750.00  1133.01  4.0
16  853.55  1053.55  4.0
17  933.01  950.00   4.0
18  982.96  829.41   4.0
19  1000.0  700.00   4.0
20  982.96  570.59   4.0
21  933.01  450.00   4.0
22  853.55  346.45   4.0
```

```
23  750.00  266.99   4.0
24  629.41  217.04   4.0
25  500.00  400.00   4.0
26  350.00  440.19   4.0
27  240.19  550.00   4.0
28  200.00  700.00   4.0
29  240.19  850.00   4.0
30  350.00  959.81   4.0
31  500.00  1000.00  4.0
32  650.00  959.81   4.0
33  759.81  850.00   4.0
34  800.00  700.00   4.0
35  759.81  550.00   4.0
36  650.00  440.19   4.0
37  500.00  700.00   4.0
38  370.59  100.00   3.5
39  500.00  100.00   3.5
40  629.41  100.00   3.5
41  370.59  0.00     3.0
42  500.00  0.00     3.0
43  629.41  0.00     3.0
1   3   41   42   38
2   3   42   43   40
3   3   42   38   39
4   3   42   40   39
5   3   38   39   1
6   3   39   40   1
7   3   38   2    1
8   3   40   24   1
9   3   1    2    25
10  3   1    24   25
11  3   24   25   36
12  3   24   23   36
13  3   23   22   36
14  3   22   36   35
15  3   22   21   35
16  3   21   20   35
17  3   20   35   34
18  3   20   19   34
19  3   19   18   34
20  3   18   34   33
21  3   18   17   33
22  3   17   16   33
23  3   16   33   32
24  3   16   15   32
25  3   15   14   32
26  3   14   32   31
27  3   14   13   31
28  3   13   12   31
29  3   12   31   30
30  3   12   11   30
31  3   11   10   30
32  3   10   30   29
33  3   10   9    29
34  3   9    8    29
35  3   8    29   28
36  3   8    7    28
37  3   7    6    28
38  3   6    28   27
39  3   6    5    27
40  3   5    4    27
41  3   4    27   26
42  3   4    3    26
43  3   3    2    26
44  3   2    26   25
45  3   26   25   37
46  3   25   36   37
47  3   36   35   37
48  3   35   34   37
49  3   34   33   37
50  3   33   32   37
```

```
51  3   32   31   37
52  3   31   30   37
53  3   30   29   37
54  3   29   28   37
55  3   28   27   37
56  3   27   26   37
1 = Number of open boundaries
3 = Total number of open boundary nodes
3 = Number of nodes for open boundary 1
41
42
43
1 = Number of land boundaries
27 = Total number of land boundary nodes
27  0 = Number of nodes for land boundary 1
43
40
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
38
41
```

===END OF FILE===

===START OF VGRID.IN FILE===

```
6 1 20.0
Z levels
1 -20.0
S levels
30. 0.7 10.
2 -1
3 -0.75
4 -0.5
5 -0.25
6 0.
```

===END OF FILE===

===START OF PARAM.IN FILE===

```
SELFE Test Run for Python Wrapper
09/10/2011 00:00:00 CST
0 ipre
0 NSCREEN
0 iwrite
0 IHOT
1  ICS
0.0 0.0  SLAM0,SFEA0
0.6 implicitness
1 0    baroclinic/barotropic
1 38 1 40
2   RNDAY
1 5
300  Dt
2  nsubfl
2 10 NDELT
1 nadv
0.01 h0
0 ntau
0.03  Cd0
0   NCOR
0. CORI
1 3600.  NWS
1 5
0 0 heat
0   turbulence closure
1.e-2 1.e-4
0 0. ihorcon
0
0. 0.
1 1  i.c.
0 isponge
0 40. ntip
0           ! NBFR
3 nope
28 -1 0 0 0
30 1 0 0 0
27 -1 0 0 0
6 1440 ! output every 30 min/new file every 30,000 min, run not this long
0   elevation: iof,touts,toutf,spool
0   Atmopheric pressure
0   Air temperature
0   Specific humidity
0   Solar radiation
0   fluxsu
0   fluxlu
0   hradu
0   hradd
0   Total flux
0   Wind speed
0   Wind stress
1   Horizontal velocity
0   Vertical velocity
0   Temperature in C
0   Salinity in psu
0   Density in kg/m^3
0   Diffusivity for transport
0   Turbulent kinetic energy
0   Turbulent mixing length
0   Test output
0 NHSTAR
1 1000 0 5.e-6 1.e-13 isolver, itmax1, iremove, zeta, tol
0 0 iflux ihcheck
1 iwmode
1 nsplit
0 ieq
```

===END OF FILE===

===START OF WIND.TH FILE===

```
0       0
1.29    0
2.50    0
3.54    0
4.33    0
4.83    0
5.00    0
4.83    0
4.33    0
3.54    0
2.50    0
1.29    0
0       0
-1.29   0
-2.50   0
-3.54   0
-4.33   0
-4.83   0
-5.00   0
-4.83   0
-4.33   0
-3.54   0
-2.50   0
-1.29   0
0       0
1.29    0
2.50    0
3.54    0
4.33    0
4.83    0
5.00    0
4.83    0
4.33    0
3.54    0
2.50    0
1.29    0
0       0
-1.29   0
-2.50   0
-3.54   0
-4.33   0
-4.83   0
-5.00   0
-4.83   0
-4.33   0
-3.54   0
-2.50   0
-1.29   0
```

===END OF FILE===

===START OF ELEV.TH FILE===

```
300     0.0065
600  0.0131
900  0.0196
1200 0.0261
1500 0.0327
1800 0.0392
2100 0.0456
2400 0.0521
2700 0.0585
3000 0.0649
3300 0.0713
3600 0.0776
3900 0.0839
4200 0.0902
4500 0.0964
4800 0.1026
5100 0.1087
5400 0.1148
5700 0.1208
6000 0.1268
6300 0.1327
6600 0.1385
6900 0.1443
7200 0.1500
7500 0.1556
7800 0.1612
8100 0.1667
8400 0.1721
8700 0.1774
9000 0.1826
9300 0.1878
9600 0.1928
9900 0.1978
10200   0.2027
10500   0.2075
10800   0.2121
11100   0.2167
11400   0.2212
11700   0.2256
12000   0.2298
12300   0.2340
12600   0.2380
12900   0.2419
13200   0.2457
13500   0.2494
13800   0.2530
14100   0.2565
14400   0.2598
14700   0.2630
15000   0.2661
15300   0.2691
15600   0.2719
15900   0.2746
16200   0.2772
16500   0.2796
16800   0.2819
17100   0.2841
17400   0.2861
17700   0.2880
18000   0.2898
18300   0.2914
18600   0.2929
18900   0.2942
19200   0.2954
19500   0.2965
19800   0.2974
20100   0.2982
20400   0.2989
20700   0.2994
```

21000	0.2997
21300	0.2999
21600	0.3000
21900	0.2999
22200	0.2997
22500	0.2994
22800	0.2989
23100	0.2982
23400	0.2974
23700	0.2965
24000	0.2954
24300	0.2942
24600	0.2929
24900	0.2914
25200	0.2898
25500	0.2880
25800	0.2861
26100	0.2841
26400	0.2819
26700	0.2796
27000	0.2772
27300	0.2746
27600	0.2719
27900	0.2691
28200	0.2661
28500	0.2630
28800	0.2598
29100	0.2565
29400	0.2530
29700	0.2494
30000	0.2457
30300	0.2419
30600	0.2380
30900	0.2340
31200	0.2298
31500	0.2256
31800	0.2212
32100	0.2167
32400	0.2121
32700	0.2075
33000	0.2027
33300	0.1978
33600	0.1928
33900	0.1878
34200	0.1826
34500	0.1774
34800	0.1721
35100	0.1667
35400	0.1612
35700	0.1556
36000	0.1500
36300	0.1443
36600	0.1385
36900	0.1327
37200	0.1268
37500	0.1208
37800	0.1148
38100	0.1087
38400	0.1026
38700	0.0964
39000	0.0902
39300	0.0839
39600	0.0776
39900	0.0713
40200	0.0649
40500	0.0585
40800	0.0521
41100	0.0456
41400	0.0392
41700	0.0327
42000	0.0261

42300	0.0196
42600	0.0131
42900	0.0065
43200	0.0000
43500	-0.0065
43800	-0.0131
44100	-0.0196
44400	-0.0261
44700	-0.0327
45000	-0.0392
45300	-0.0456
45600	-0.0521
45900	-0.0585
46200	-0.0649
46500	-0.0713
46800	-0.0776
47100	-0.0839
47400	-0.0902
47700	-0.0964
48000	-0.1026
48300	-0.1087
48600	-0.1148
48900	-0.1208
49200	-0.1268
49500	-0.1327
49800	-0.1385
50100	-0.1443
50400	-0.1500
50700	-0.1556
51000	-0.1612
51300	-0.1667
51600	-0.1721
51900	-0.1774
52200	-0.1826
52500	-0.1878
52800	-0.1928
53100	-0.1978
53400	-0.2027
53700	-0.2075
54000	-0.2121
54300	-0.2167
54600	-0.2212
54900	-0.2256
55200	-0.2298
55500	-0.2340
55800	-0.2380
56100	-0.2419
56400	-0.2457
56700	-0.2494
57000	-0.2530
57300	-0.2565
57600	-0.2598
57900	-0.2630
58200	-0.2661
58500	-0.2691
58800	-0.2719
59100	-0.2746
59400	-0.2772
59700	-0.2796
60000	-0.2819
60300	-0.2841
60600	-0.2861
60900	-0.2880
61200	-0.2898
61500	-0.2914
61800	-0.2929
62100	-0.2942
62400	-0.2954
62700	-0.2965
63000	-0.2974
63300	-0.2982

63600	-0.2989
63900	-0.2994
64200	-0.2997
64500	-0.2999
64800	-0.3000
65100	-0.2999
65400	-0.2997
65700	-0.2994
66000	-0.2989
66300	-0.2982
66600	-0.2974
66900	-0.2965
67200	-0.2954
67500	-0.2942
67800	-0.2929
68100	-0.2914
68400	-0.2898
68700	-0.2880
69000	-0.2861
69300	-0.2841
69600	-0.2819
69900	-0.2796
70200	-0.2772
70500	-0.2746
70800	-0.2719
71100	-0.2691
71400	-0.2661
71700	-0.2630
72000	-0.2598
72300	-0.2565
72600	-0.2530
72900	-0.2494
73200	-0.2457
73500	-0.2419
73800	-0.2380
74100	-0.2340
74400	-0.2298
74700	-0.2256
75000	-0.2212
75300	-0.2167
75600	-0.2121
75900	-0.2075
76200	-0.2027
76500	-0.1978
76800	-0.1928
77100	-0.1878
77400	-0.1826
77700	-0.1774
78000	-0.1721
78300	-0.1667
78600	-0.1612
78900	-0.1556
79200	-0.1500
79500	-0.1443
79800	-0.1385
80100	-0.1327
80400	-0.1268
80700	-0.1208
81000	-0.1148
81300	-0.1087
81600	-0.1026
81900	-0.0964
82200	-0.0902
82500	-0.0839
82800	-0.0776
83100	-0.0713
83400	-0.0649
83700	-0.0585
84000	-0.0521
84300	-0.0456
84600	-0.0392

84900	-0.0327
85200	-0.0261
85500	-0.0196
85800	-0.0131
86100	-0.0065
86400	0.0000
86700	0.0065
87000	0.0131
87300	0.0196
87600	0.0261
87900	0.0327
88200	0.0392
88500	0.0456
88800	0.0521
89100	0.0585
89400	0.0649
89700	0.0713
90000	0.0776
90300	0.0839
90600	0.0902
90900	0.0964
91200	0.1026
91500	0.1087
91800	0.1148
92100	0.1208
92400	0.1268
92700	0.1327
93000	0.1385
93300	0.1443
93600	0.1500
93900	0.1556
94200	0.1612
94500	0.1667
94800	0.1721
95100	0.1774
95400	0.1826
95700	0.1878
96000	0.1928
96300	0.1978
96600	0.2027
96900	0.2075
97200	0.2121
97500	0.2167
97800	0.2212
98100	0.2256
98400	0.2298
98700	0.2340
99000	0.2380
99300	0.2419
99600	0.2457
99900	0.2494
100200	0.2530
100500	0.2565
100800	0.2598
101100	0.2630
101400	0.2661
101700	0.2691
102000	0.2719
102300	0.2746
102600	0.2772
102900	0.2796
103200	0.2819
103500	0.2841
103800	0.2861
104100	0.2880
104400	0.2898
104700	0.2914
105000	0.2929
105300	0.2942
105600	0.2954
105900	0.2965

106200	0.2974
106500	0.2982
106800	0.2989
107100	0.2994
107400	0.2997
107700	0.2999
108000	0.3000
108300	0.2999
108600	0.2997
108900	0.2994
109200	0.2989
109500	0.2982
109800	0.2974
110100	0.2965
110400	0.2954
110700	0.2942
111000	0.2929
111300	0.2914
111600	0.2898
111900	0.2880
112200	0.2861
112500	0.2841
112800	0.2819
113100	0.2796
113400	0.2772
113700	0.2746
114000	0.2719
114300	0.2691
114600	0.2661
114900	0.2630
115200	0.2598
115500	0.2565
115800	0.2530
116100	0.2494
116400	0.2457
116700	0.2419
117000	0.2380
117300	0.2340
117600	0.2298
117900	0.2256
118200	0.2212
118500	0.2167
118800	0.2121
119100	0.2075
119400	0.2027
119700	0.1978
120000	0.1928
120300	0.1878
120600	0.1826
120900	0.1774
121200	0.1721
121500	0.1667
121800	0.1612
122100	0.1556
122400	0.1500
122700	0.1443
123000	0.1385
123300	0.1327
123600	0.1268
123900	0.1208
124200	0.1148
124500	0.1087
124800	0.1026
125100	0.0964
125400	0.0902
125700	0.0839
126000	0.0776
126300	0.0713
126600	0.0649
126900	0.0585
127200	0.0521

127500	0.0456
127800	0.0392
128100	0.0327
128400	0.0261
128700	0.0196
129000	0.0131
129300	0.0065
129600	0.0000
129900	-0.0065
130200	-0.0131
130500	-0.0196
130800	-0.0261
131100	-0.0327
131400	-0.0392
131700	-0.0456
132000	-0.0521
132300	-0.0585
132600	-0.0649
132900	-0.0713
133200	-0.0776
133500	-0.0839
133800	-0.0902
134100	-0.0964
134400	-0.1026
134700	-0.1087
135000	-0.1148
135300	-0.1208
135600	-0.1268
135900	-0.1327
136200	-0.1385
136500	-0.1443
136800	-0.1500
137100	-0.1556
137400	-0.1612
137700	-0.1667
138000	-0.1721
138300	-0.1774
138600	-0.1826
138900	-0.1878
139200	-0.1928
139500	-0.1978
139800	-0.2027
140100	-0.2075
140400	-0.2121
140700	-0.2167
141000	-0.2212
141300	-0.2256
141600	-0.2298
141900	-0.2340
142200	-0.2380
142500	-0.2419
142800	-0.2457
143100	-0.2494
143400	-0.2530
143700	-0.2565
144000	-0.2598
144300	-0.2630
144600	-0.2661
144900	-0.2691
145200	-0.2719
145500	-0.2746
145800	-0.2772
146100	-0.2796
146400	-0.2819
146700	-0.2841
147000	-0.2861
147300	-0.2880
147600	-0.2898
147900	-0.2914
148200	-0.2929
148500	-0.2942

148800	-0.2954
149100	-0.2965
149400	-0.2974
149700	-0.2982
150000	-0.2989
150300	-0.2994
150600	-0.2997
150900	-0.2999
151200	-0.3000
151500	-0.2999
151800	-0.2997
152100	-0.2994
152400	-0.2989
152700	-0.2982
153000	-0.2974
153300	-0.2965
153600	-0.2954
153900	-0.2942
154200	-0.2929
154500	-0.2914
154800	-0.2898
155100	-0.2880
155400	-0.2861
155700	-0.2841
156000	-0.2819
156300	-0.2796
156600	-0.2772
156900	-0.2746
157200	-0.2719
157500	-0.2691
157800	-0.2661
158100	-0.2630
158400	-0.2598
158700	-0.2565
159000	-0.2530
159300	-0.2494
159600	-0.2457
159900	-0.2419
160200	-0.2380
160500	-0.2340
160800	-0.2298
161100	-0.2256
161400	-0.2212
161700	-0.2167
162000	-0.2121
162300	-0.2075
162600	-0.2027
162900	-0.1978
163200	-0.1928
163500	-0.1878
163800	-0.1826
164100	-0.1774
164400	-0.1721
164700	-0.1667
165000	-0.1612
165300	-0.1556
165600	-0.1500
165900	-0.1443
166200	-0.1385
166500	-0.1327
166800	-0.1268
167100	-0.1208
167400	-0.1148
167700	-0.1087
168000	-0.1026
168300	-0.0964
168600	-0.0902
168900	-0.0839
169200	-0.0776
169500	-0.0713
169800	-0.0649

```
170100  -0.0585
170400  -0.0521
170700  -0.0456
171000  -0.0392
171300  -0.0327
171600  -0.0261
171900  -0.0196
172200  -0.0131
172500  -0.0065
172800  0.0000
```

===END OF FILE===

===START OF GNOME BNA MAP FILE===

```
GNOME Map File
SELFE Example
2011 9 11 00 00 00

1
25
40  629.41  100.00
24  629.41  217.04
23  750.00  266.99
22  853.55  346.45
21  933.01  450.00
20  982.96  570.59
19  1000.0  700.00
18  982.96  829.41
17  933.01  950.00
16  853.55  1053.55
15  750.00  1133.01
14  629.41  1182.96
13  500.00  1200.00
12  370.59  1182.96
11  250.00  1133.01
10  146.45  1053.55
9   66.99   950.00
8   17.04   829.41
7   0.00    700.00
6   17.04   570.59
5   66.99   450.00
4   146.45  346.45
3   250.00  266.99
2   370.59  217.04
38  370.59  100.00
```

===END OF FILE===

===START OF GNOME WIND.WND FILE===

```
11,9,2011,01,00,00,0.00,W
11,9,2011,02,00,00,1.29,W
11,9,2011,03,00,00,2.50,W
11,9,2011,04,00,00,3.54,W
11,9,2011,05,00,00,4.33,W
11,9,2011,06,00,00,4.83,W
11,9,2011,07,00,00,5.00,W
11,9,2011,08,00,00,4.83,W
11,9,2011,09,00,00,4.33,W
11,9,2011,10,00,00,3.54,W
11,9,2011,11,00,00,2.50,W
11,9,2011,12,00,00,1.29,W
11,9,2011,13,00,00,0.00,W
11,9,2011,14,00,00,-1.29,W
11,9,2011,15,00,00,-2.50,W
11,9,2011,16,00,00,-3.54,W
11,9,2011,17,00,00,-4.33,W
11,9,2011,18,00,00,-4.83,W
11,9,2011,19,00,00,-5.00,W
11,9,2011,20,00,00,-4.83,W
11,9,2011,21,00,00,-4.33,W
11,9,2011,22,00,00,-3.54,W
11,9,2011,23,00,00,-2.50,W
11,9,2011,24,00,00,-1.29,W
12,9,2011,01,00,00,0.00,W
12,9,2011,02,00,00,1.29,W
12,9,2011,03,00,00,2.50,W
12,9,2011,04,00,00,3.54,W
12,9,2011,05,00,00,4.33,W
12,9,2011,06,00,00,4.83,W
12,9,2011,07,00,00,5.00,W
12,9,2011,08,00,00,4.83,W
12,9,2011,09,00,00,4.33,W
12,9,2011,10,00,00,3.54,W
12,9,2011,11,00,00,2.50,W
12,9,2011,12,00,00,1.29,W
12,9,2011,13,00,00,0.00,W
12,9,2011,14,00,00,-1.29,W
12,9,2011,15,00,00,-2.50,W
12,9,2011,16,00,00,-3.54,W
12,9,2011,17,00,00,-4.33,W
12,9,2011,18,00,00,-4.83,W
12,9,2011,19,00,00,-5.00,W
12,9,2011,20,00,00,-4.83,W
12,9,2011,21,00,00,-4.33,W
12,9,2011,22,00,00,-3.54,W
12,9,2011,23,00,00,-2.50,W
12,9,2011,24,00,00,-1.29,W
```

===END OF FILE===

===START OF GNOME COMMAND FILE===

```
# Note: The lines of this file are wrapped to fit on a page. In the   # actual file, each MESSAGE should be one long line.

[GNOME COMMAND FILE]

MESSAGE createMap; TO model; TYPE vector; NAME TestMap; PATH \full\path\to\testmap.bna;

MESSAGE createMover; TO TestMap; TYPE ptCur; NAME ModelCurrents; PATH \full\path\to\gnome_in_currents.txt;

MESSAGE createMover; TO Universal Map; TYPE Wind; NAME SineWind; PATH \full\path\to\wind.ossm; speedUnits mph;

MESSAGE createSpill; TO model; NAME TestSpill; startRelTime 11,09,2011,00,00; numLEs 1000; startRelPos Lat/Long;

Message run; TO model; startTime 11,09,2011,00,00; runDurationInHrs 48; timeStepInMinutes 30; outputStepInMinutes 30; outputFolder :GnomeOut:;
```

===END OF FILE===

C References

ASCE Task Committee on Modeling of Oil Spills of the Water Resources Engineering Division (1996). “State-of-the-art review of modeling transport and fate of oil spills.” *Journal of Hydraulic Engineering-ASCE* 122(11): 594-609.

Başar, E., Köse, E., and Güneroglu, A. (2006). “Finding risky areas for oil spillage after tanker accidents at Istanbul Strait.” *International Journal of Environment and Pollution* 27(4): 388–400.

Beegle-Krause, C.J. (2001). “General NOAA Oil Modeling Environment (GNOME): A New Spill Trajectory Model.” *IOSC 2001 Proceedings* 2:865-871.

Black, T. L. (1994). “The new NMC mesoscale Eta Model: Description and forecast examples.” *Weather Forecasting* 9: 265-278.

Bulatewicz, T., Yang, X., Peterson, J.M., Staggenborg, S., Welch, S.M., and Steward, D.R. (2010). “Accessible integration of agriculture, groundwater, and economic models using the Open Modeling Interface (OpenMI): methodology and initial results.” *Hydrology and Earth System Sciences* 14: 521-534.

Christensen, F.D. (2004). “Coupling Between the River Basin Management Model (MIKE BASIN) and the 3D Hydrological Model (MIKE SHE) with the use of the OpenMI system.” in Liong, S.-Y, K.-K. Phoon, and V. Babovic (eds.), *Proceedings of the 6th International Conference on Hydroinformatics*. Singapore, 21-24 June 2004, World Scientific Publishing Co. 2076 pgs.

Crockett, D. (2010). “The Texas Water Development Board: Hydrodynamic and Oil Spill Modeling” Texas Water Development Board website: <http://midgewater.twdb.state.tx.us/bays_estuaries/bhydpage.html>.

Department of Commerce (1998). “Texas Coast (G200) Bathymetric Digital Elevation Model (30 Meter Resolution).” NOAA Ocean Service Special Projects Online Archive: <http://estuarinebathymetry.noaa.gov>.

Fach, B.A., and Klinck, J.M. (2006). “Transport of Antarctic krill (Euphausia superba) across the Scotia Sea. Part I: Circulation and particle tracking simulations.” *Deep Sea Research Part I: Oceanographic Research Papers* 53(6): 987-1010.

Furnans, J. (2004). *Exploring Hydrodynamic Modeling of Texas Bays with focus on Corpus Christi Bay & Lavaca Bay*.” CRWR Online Report: 04-3, 30 pgs. <http://www.crwr.utexas.edu/online.shtml>.

Gong, W., J. Shen, K.H. Cho, and H.V. Wang (2009). “A numerical model study of barotropic subtidal water exchange between estuary and subestuaries (tributaries) in the Chesapeake Bay during northeaster events.” *Ocean Modelling* 26(3-4):170-189.

Kerbaol, V., and F. Collard (2005). “SAR-derived coastal and marine applications: From research to operational products.” *IEEE Journal of Oceanic Engineering* 30(3): 472-486.

Kulis, P.S. and B. R. Hodges (2006), “Modeling gravity currents in shallow bays using a sigma coordinate model.” *Proceedings of the 7th International Conference on Hydroscience and Engineering*, Philadelphia, PA, Electronic Proceedings: <http://hdl.handle.net/1860/1458>.

Lutz, M. (2011). *Programming Python*, 4th Edition. O’Rielly Media, Sebastopol California, USA. 1552 pgs.

Min, D. (2010). “Characteristics of ocean currents and water properties observed throughout the water column at the Aransas Pass Tidal Inlet during Summer/Fall 2010.” Presentation at the Gulf Estuarine Research Society Meeting, Nov. 3-4, 2010, Port Aransas, Texas, USA.

Mitchell, J.C. (2003). *Concepts in Programming Languages.* Cambridge University Press. Cambridge, UK. 529 pgs.

Mueller, J., Wagner, A., Fahl, K. et al. (2011). “Towards quantitative sea ice reconstructions in the northern North Atlantic: A combined biomarker and numerical modelling approach.” Earth and Planetary Science Letters 306(3-4): 137-148.

NOAA OR&R (2011) “Office of Response and Restoration: Responding to Oil Spills.” NOAA Office of Response and Restoration website: <http://response.restoration.noaa.gov/>.

Oliveira, A., A.B. Fortunato, and L. Pinto (2006). “Modeling the Hydrodynamics and the Fate of Passive and Active Tracers in the Guadiana Estuary.” *Estuarine, Coastal and Shelf Science* 70: 76-84.

OpenMI Association (2011). “What is OpenMI?” OpenMI Association website. <http://www.openmi.org>.

Rosenzweig, I. (2011). *Integrating hydrodynamic and oil spill trajectory models for nowcasts/forecasts of Texas bays*. M.S. Thesis, Department of Civil, Architectural, and Environmental Engineering, University of Texas at Austin, 41 pgs. http://hdl.handle.net/2152/ETD-UT-2011-08-4102

Port of Houston Authority (2011). “General Information: The Port of Houston.” Port of Houston Authority website: < http://www.portofhouston.com >.

Pothina, D. (2009) *A multimodel approach to modeling bay circulation in shallow bay - ship channel systems*. Ph.D. dissertation, Department of Civil, Architectural, and Environmental Engineering, University of Texas at Austin, 128 pgs.

Powell, G.L., J. Matsumoto, W.L. Longley, and D.A. Brock (1997). *Effects of Structures and Practices on the Circulation and Salinity Patterns of the Corpus Christi Bay National Estuary Program Area, Texas*. Corpus Christi Bay National Estuary Program Technical Report CCBNEP-19, October 1997, 169 pgs. Available at: http://www.cbbep.org/publicationsPM.html

Rodrigues, M., A. Oliveira, H. Queiroga, A.B. Fortunato, and Y.J. Zhang (2009). "Three-Dimensional Modeling of the Lower Trophic Levels in the Ria de Aveiro (Portugal)." *Ecological Modelling* 220: 1274-1290.

TAMU Department of Atmospheric Sciences (2010). "Weather Interface." Texas A&M University Department of Atmospheric Science website: <http://leonardo.met.tamu.edu/Weather_Interface>. Accessed: Apr. 8, 2010.

Umlauf, L., and H. Burchard (2003)." A generic length-scale equation for geophysical turbulence models." *Journal of Marine Research* 6: 235–265.

US Department of Commerce (2011). "National Data Buoy Center." NDBC website: <http://www.ndbc.noaa.gov/>.

Wang, S.D., Y.M. Shen, and Y.H. Zheng (2005). "Two-dimensional numerical simulation for transport and fate of oil spills in seas." *Ocean Engineering* 32(13): 1556-1571.

Wang, J.H., and Y.M. Shen (2010). "Development of an integrated model system to simulate transport and fate of oil spills in seas." *Science China – Technological Sciences* 53(9): 2423-2434.

Zhang, Y.-L. and A.M. Baptista, (2008). "SELFE: A semi-implicit Eulerian-Lagrangian finite-element model for cross-scale ocean circulation." *Ocean Modeling* 21(3-4): 71-96.

D List of publications developed from this research

This report: Rosenzweig, I. and B.R. Hodges (2011). *A Python Wrapper for Coupling Hydrodynamic and Oil Spill Models*. CRWR Online Report 11-09, Center for Research in Water Resources, University of Texas at Austin, 42 pgs. http://www.crwr.utexas.edu/online.shtml

Rosenzweig, I. (2011). *Integrating hydrodynamic and oil spill trajectory models for nowcasts/forecasts of Texas bays*. M.S. Thesis, Department of Civil, Architectural, and Environmental Engineering, University of Texas at Austin, 41 pgs. http://hdl.handle.net/2152/ETD-UT-2011-08-4102

Rosenzweig, I., and B.R. Hodges (2011). "Integrating hydrodynamic and oil spill trajectory models for nowcasts/forecasts of Texas bays." Poster session presented at Gordon Research Conference - Coastal Ocean Modeling; June 26 - July 1, 2011; Mount Holyoke, Massachusetts.

Rosenzweig, I. and B.R. Hodges (2011). "Integrating hydrodynamic and oil spill trajectory models for nowcasts/forecasts of Texas bays." International Oil Spill Conference, Portland, Oregon, May 23-26, 2011.

Rosenzweig, I. and B.R. Hodges (2010). "Integrating hydrodynamic and oil spill trajectory models for nowcasts/forecasts of Texas bays." 2010 Biennial Gulf Estuarine Research Society Meeting, Port Aransas, Texas, Nov. 3-4. 2010.

finis